Toward Interoperable Grid Infrastructure Software: Basing Resource and User Authentication in Legion on GSI

Marty Humphrey

University of Virginia

Department of Computer Science

http://legion.virginia.edu

Outline

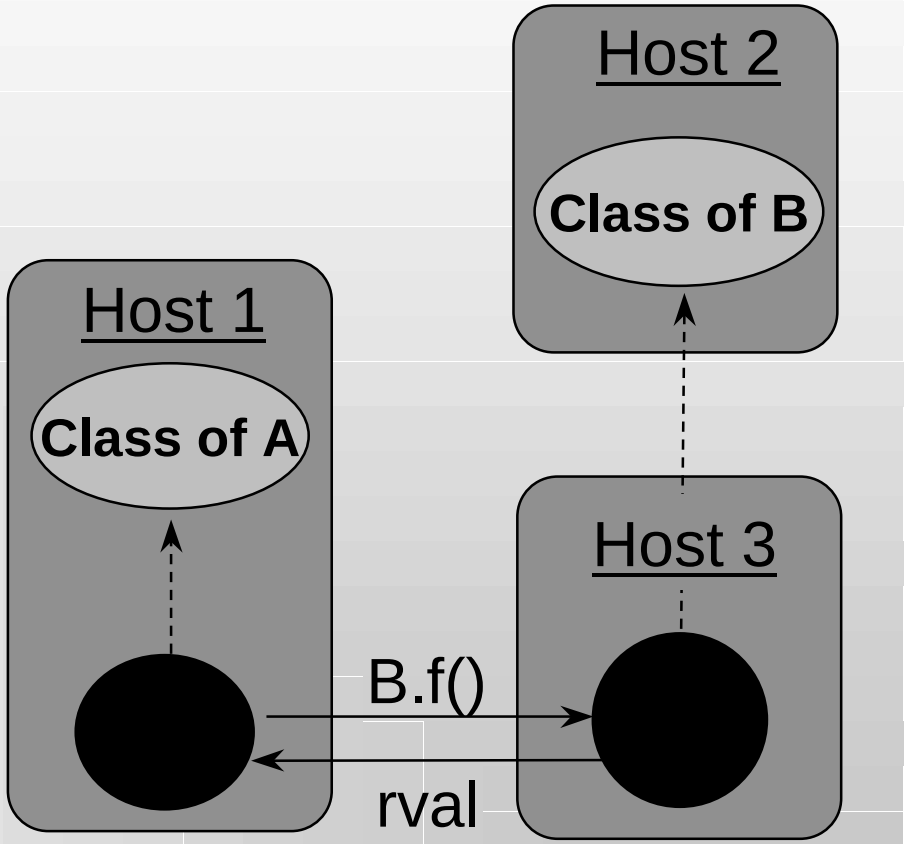
- Brief review of Legion
- Legion security
- GSI
- Legion over GSI
  - Successes
  - Limitations of GSI
  - Future
- Summary

Legion: a Distributed Meta-Operating System

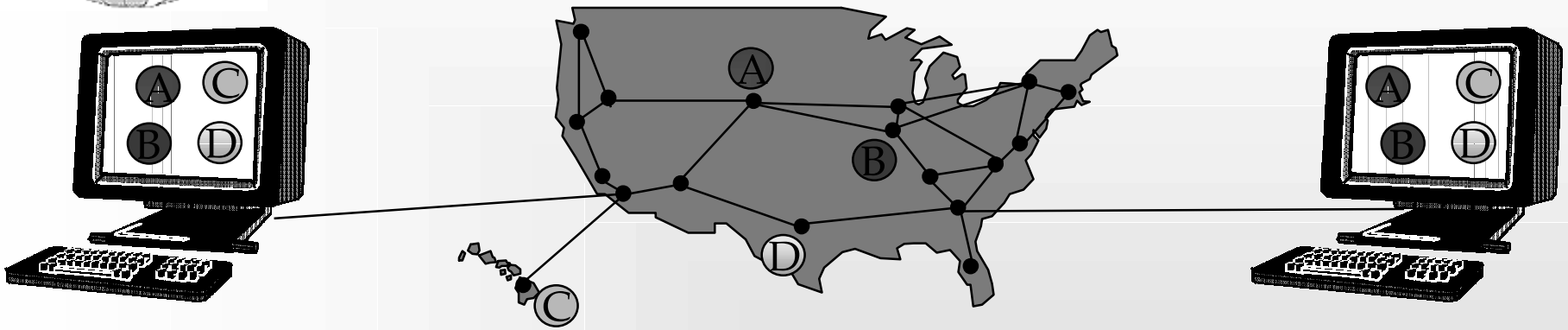
- Provides services of a traditional OS
  - Process creation and control
  - File system
  - Security
  - Resource management and accounting
- Runs on top of existing OSs
  - Local systems provide the raw materials
  - Legion combines the resources into a single system (single virtual machine)
- Current version: Legion 1.7

The Object Model

- Legion is **object-based** (does not mean C++)
- Legion **objects**:
  - belong to classes
  - are logically independent, address-space-disjoint
  - communicate via non-blocking method calls
  - are *Active* or *Inert*



Shared Persistent Spaces

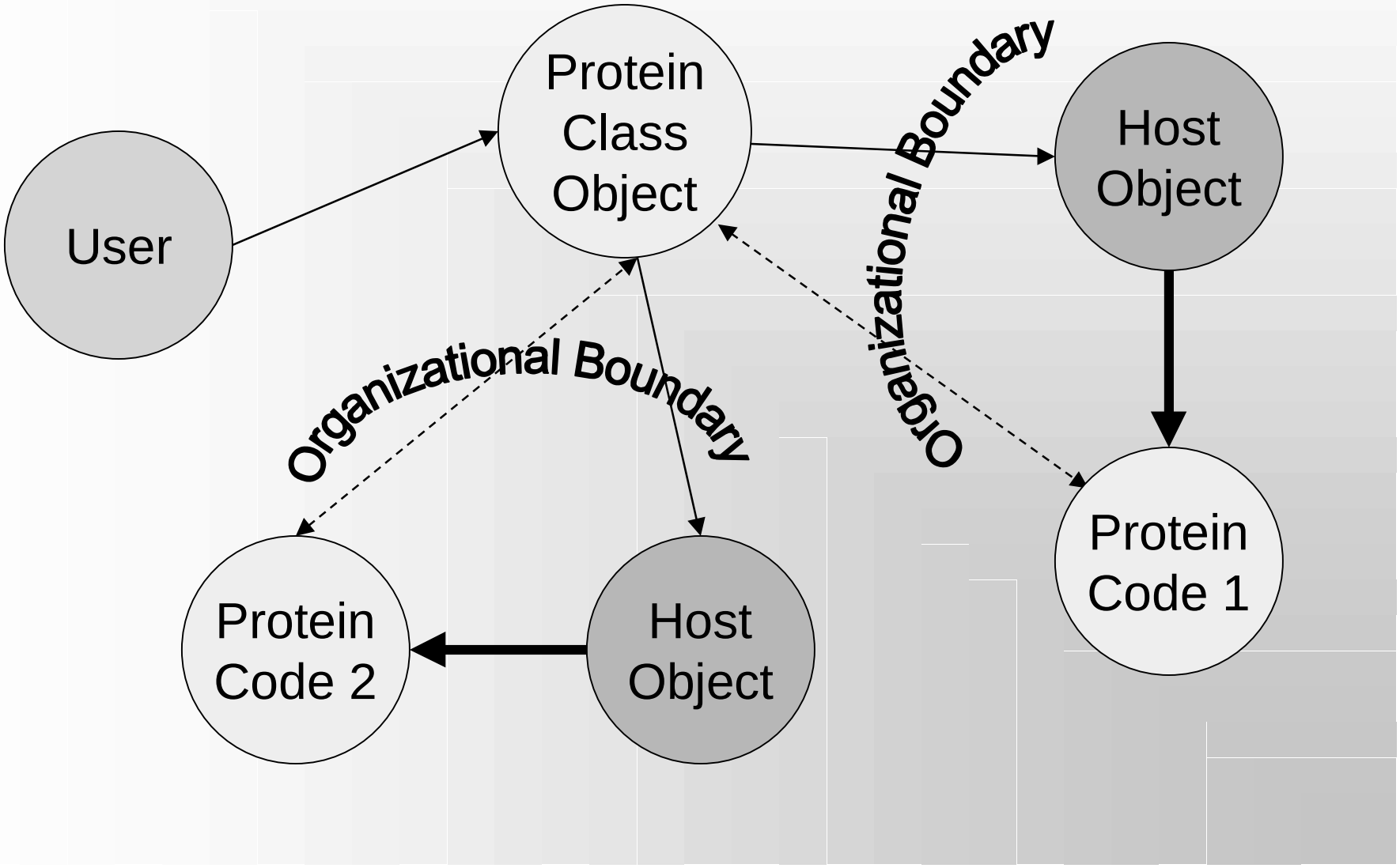


- Location transparent access to data and executing objects (e.g., simulations)
- Single name space
- Usual Unix utilities, legion_ls etc.
- Files are objects - application specific interfaces
  - Parallel file objects
  - Manage data format heterogeneity
- C, C++, Fortran, Java interfaces - stdio and stream calls

Executing Computations

- *legion_run*, *legion_run_multi* - support both legacy and Legion aware codes
- Legion selects site, move binaries, monitor progress, etc.
- Parallel processing
  - Bag of tasks
  - Native MPI, Legion MPI, PVM, parallel C++, Fortran (coarse-grain)

Executing an Application (Starting an Object)



Some Legion Capabilities

- Fault tolerance (e.g., object replication)
- Collaboration via object “sharing”
  - Active simulations; Datasets
- Debugging via message replay
- General event management facilities
  - Generation and publish events; Selective subscription to events

General Grid Security

- Require standard security services
  - e.g., confidentiality, authentication, access control
- But: cross-domain
  - No “super-user”
  - No code-base trusted by all sites
- And: large-scale
  - Must assume some compromised hosts
  - Require site isolation
- Legion and Globus largely agree on:
  - Basic model
  - Public-key cryptography as enabling technology

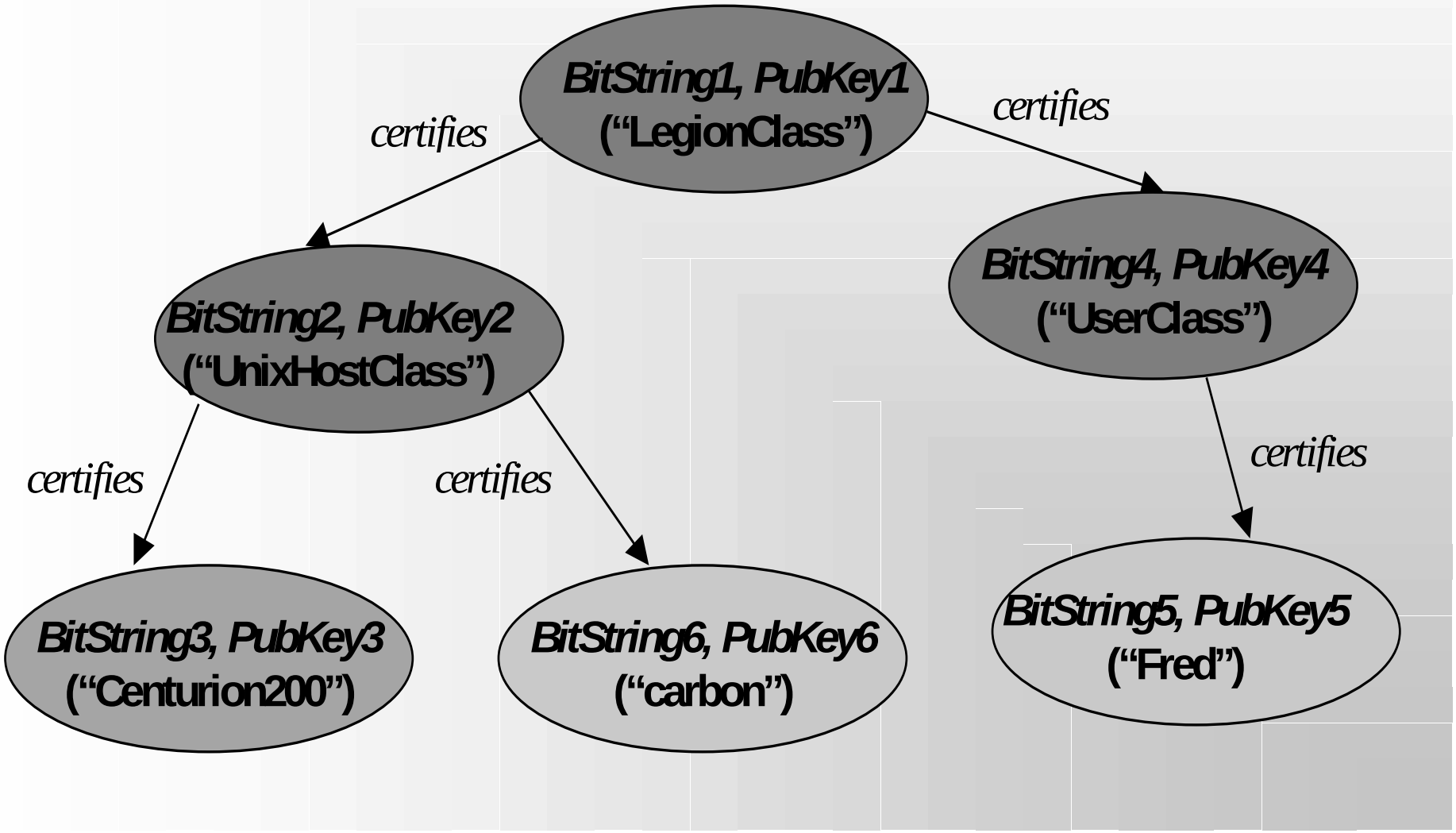
Key Differences Between Legion and Globus Security

- How are public keys certified?
- How is mutual authentication achieved?
- How is authorization performed?
- How is restricted delegation achieved?

Legion Identity

- Based on Legion extensible object naming mechanism
- Every object has an RSA public key as part of its name
  - Instead of just Public Key, it could be X.509 cert
- “Name” of object is inseparable from Public Key

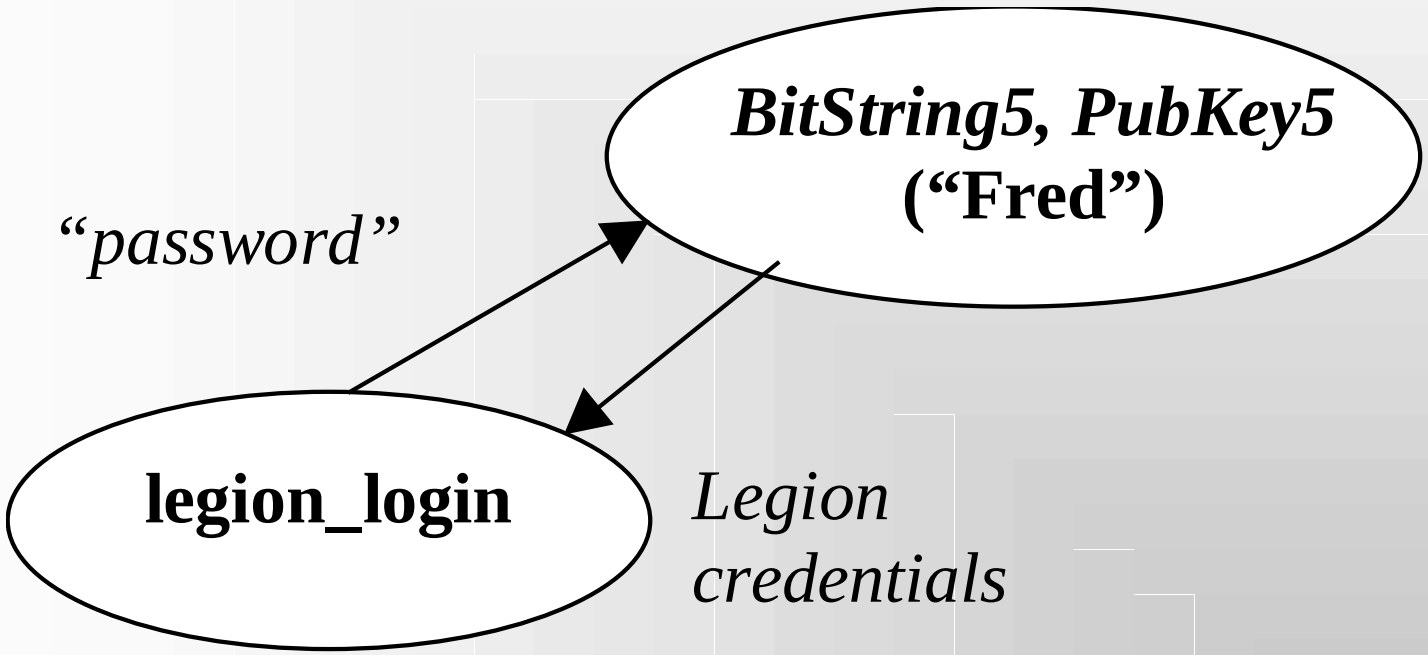
Legion Certification Structure



Legion Certification

- A Certificate Authority should not be *required* (but can be supported)
- For *some* installations, CA is unnecessary
  - Why should some unknown entity have to be trusted?
  - Why wait for a CA to get around to responding with the certification?
- Certification is through *Web of Trust*
  - Though rooted in *LegionClass*

Legion Login

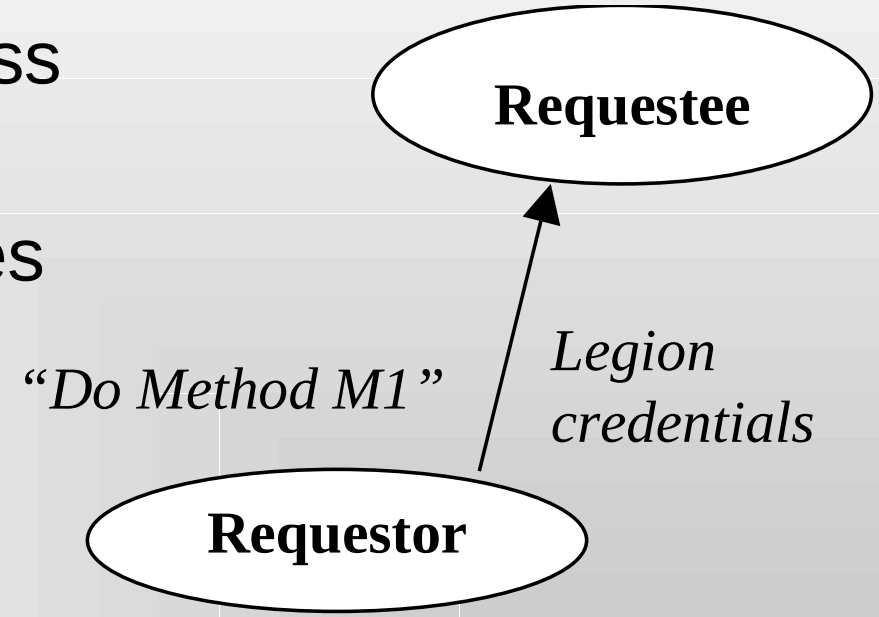


Legion Credentials

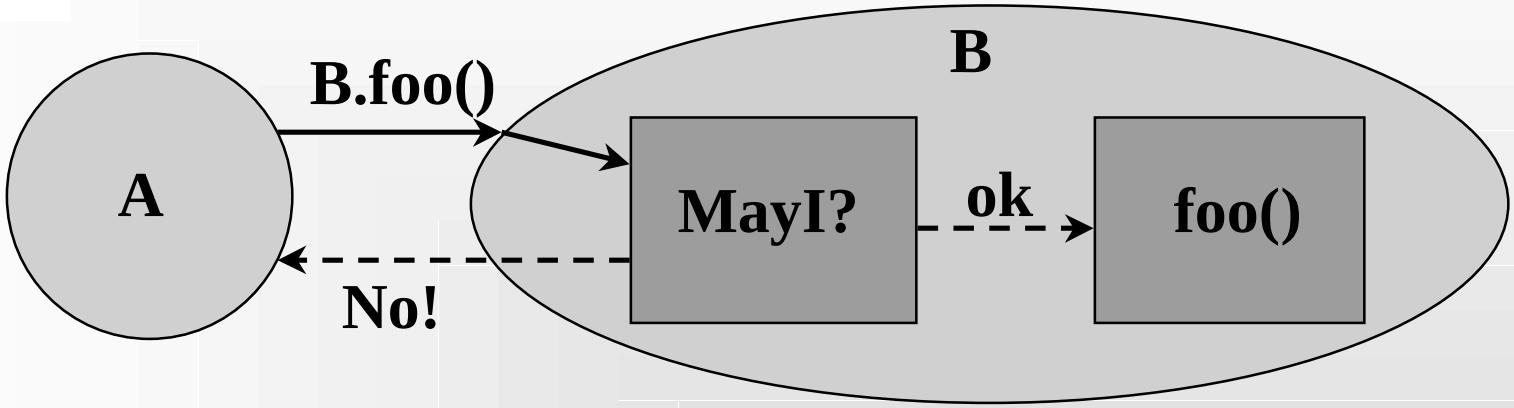
- Single credential used for both identity and authorization:
  - Object or class restrictions
  - Method restrictions
  - Time restrictions
- Possession of multiple credentials imply multiple roles
- “Login” returns unrestricted credential:
  - *[The bearer of this credential has all the rights of Alice] signed Alice’s “Authentication Object”*
- Legion credentials fully support *restricted delegation*

Mutual Authentication

- Requestor authenticates Requestee through Class Object of Requestee
- Requestee authenticates Requestor through:
  - Class of Requestor
  - Legion credentials are signed by requestor's private key



Access Control



- Access policy locally defined and evaluated
  - Method-by-method basis - MayI
  - Use a standard policy or *define your own*
    - Default: per-method ACLs based on users, groups
  - Authorization based on credentials
    - Delegated (restricted) credentials or bearer credentials

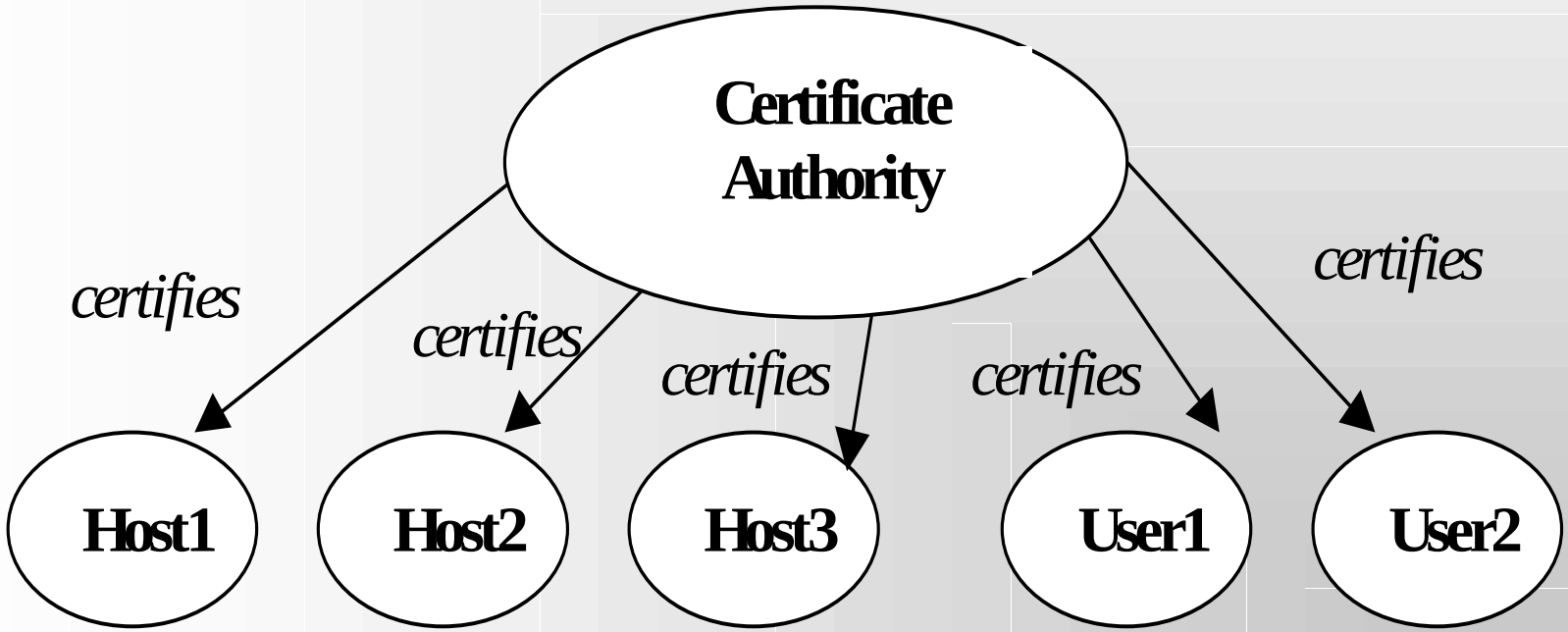
Some Legion Security Capabilities

- "All communication in the Grid due to my scheduling request should be encrypted."
- "I want to allow Alice to view the output of my executing job in real-time."
- "I don't care where you store my temporary files associated with my job, but they must be encrypted on secondary storage."
- "Schedule my job anywhere except *host1, host2,* and *host15,* because I don't trust them."

Globus Security Infrastructure (GSI)

- OpenSSl (*ssleay)* provides many crypto routines and security-related mechanisms such as:
  - RSA, DES, Triple-DES, X.509
- GSI augments OpenSSL with:
  - GSSAPI binding to SSL
  - Delegation (impersonation)
  - Some credential management routines

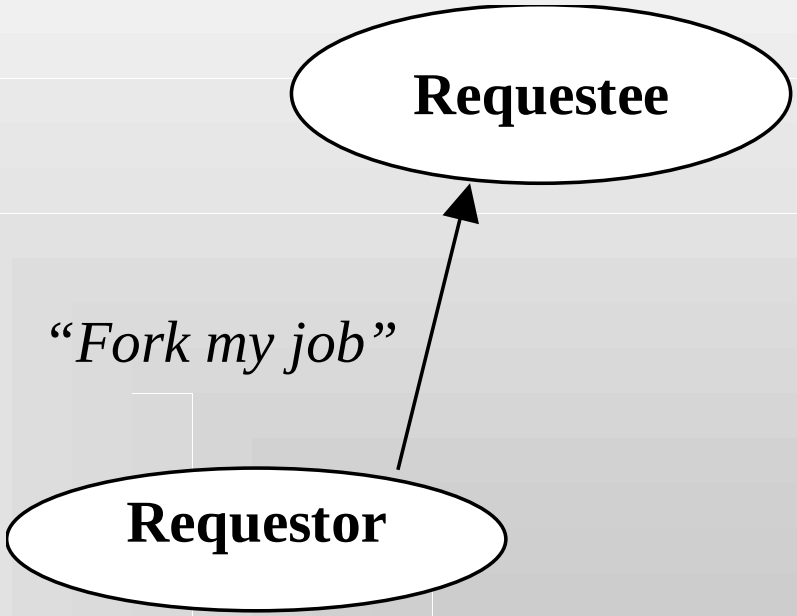
GSI Certification Structure



GSI Mutual Authentication

- GSSAPI
- SSL
- Certificate Authority

- Credential Forwarding (“delegation” via GSSAPI/X.509)



Why run Legion over GSI?

- CA arguably “more secure” than implicit trust in *LegionClass*
- Legion components and Globus components can directly interact
  - E.g., a Legion scheduling component can access the Globus LDAP server
- But there are some downsides...

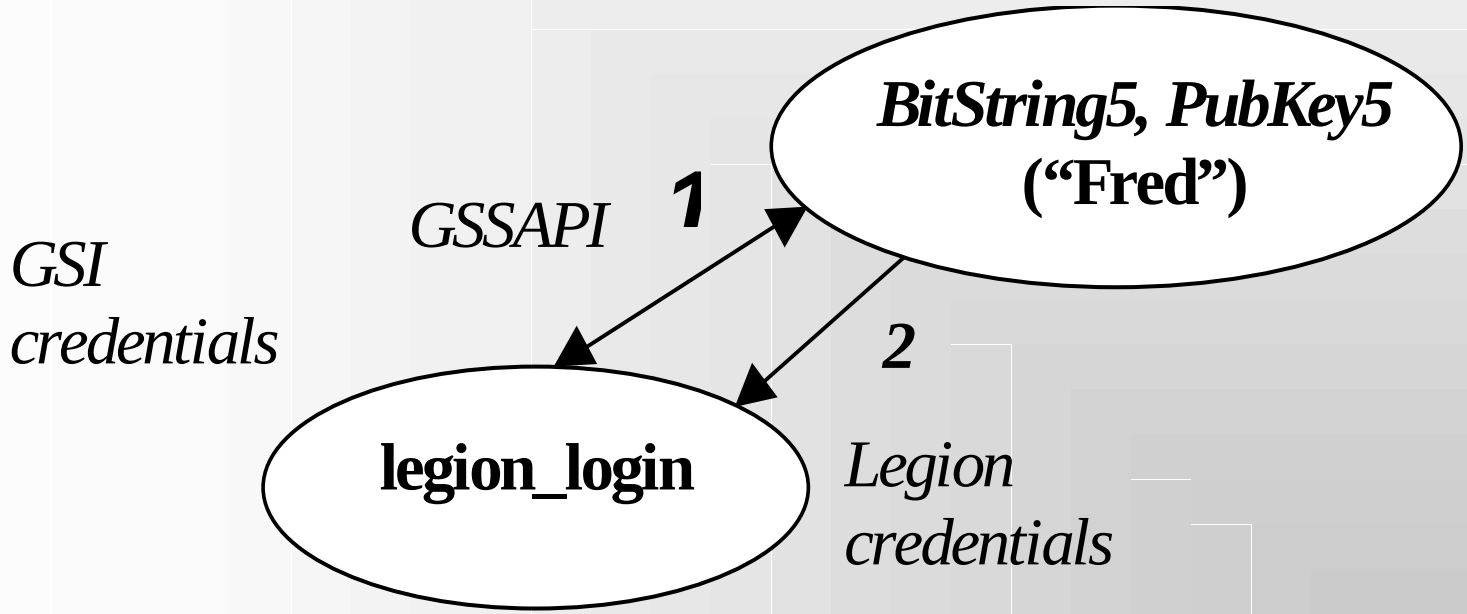
GSI Limitations

- Is mutual authentication via SSL always appropriate?
- What about delegation?
  - GSSAPI/X.509 may ***impede*** restricted delegation (it is an *authentication and message privacy API, not a delegation/authorization API)*
  - Authorization info in X.509?
- What about access control?
  - Does GSI mandate GAA? Does GAA add value over Legion's Access Control?

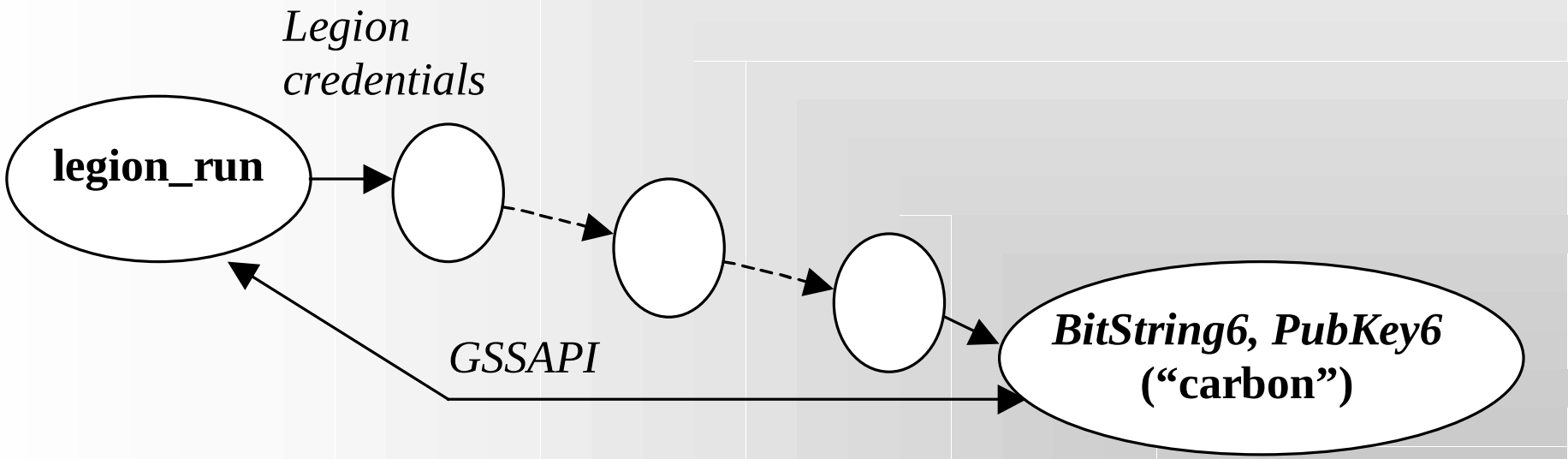
Legion over GSI: Current Approach

- Base “login” to Legion on valid GSI creds
  - Don’t require specific password for both Legion and Globus
- Base process creation on valid GSI creds
  - Arguably, a stronger sense of security
- Use Legion credentials everywhere else
  - Retain delegation capability
  - Retain Access Control capability

Legion Login via GSI



Authentication to Legion Host Object via GSI



Security Standards

- Value of using GSSAPI in Legion?
- Legion *user* does not have to deal with GSSAPI ugliness:
  - Authentication is part of Legion infrastructure
  - Access Control is part of the Legion infrastructure
  - --> Writer of SuperScheduler in Legion can focus on scheduling issues, not security calls

Summary

- GSI has been utilized in Legion for:
  - GSI-based session establishment (“login”)
  - Enhanced security at time of process creation
- Issue: What does additional use of GSI provide to:
  - Average Legion Grid user?
  - Legion installer and maintainer?
- Delegation capability is not as far along in GSI as we need in Legion